INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Numerical Scheme for the One-Dimensional Vlasov-Poisson Equation Using Bi-Orthogonal Spline Wavelets

Vincent Torri

No 5757

Novembre 2005

Thème NUM

ISSN 0249-6399

Numerical Scheme for the One-Dimensional Vlasov-Poisson Equation Using Bi-Orthogonal Spline Wavelets

Vincent Torri*

Thème NUM — Systèmes numériques
Projet Calvi

Rapport de recherche no 5757 — Novembre 2005 — 26 pages

Abstract: In this paper, a numerical scheme for solving the Vlasov-Poisson equations is proposed. It is based on bi-orthogonal compactly supported spline wavelets. The interest of these wavelets used in this method is their precision in computations. For solving the Vlasov equation, a Strang splitting in time and a semi-lagrangian method are used. For the Poisson equation, a solver based only on wavelets is presented.

Key-words: Vlasov-Poisson equation, bi-orthogonal wavelets, semi-lagrangian scheme

* Université d'Évry Val-d'Essonne, Bâtiment Maupertuis, 91025 Évry Cédex - FRANCE. E-mail: vtorri@univ-evry.fr

Unité de recherche INRIA Lorraine
LORIA, Technopôle de Nancy-Brabois, Campus scientifique,
615, rue du Jardin Botanique, BP 101, 54602 Villers-Lès-Nancy (France)
Téléphone : +33 3 83 59 30 00 — Télécopie : +33 3 83 27 83 19

Schéma numérique pour l'équation de Vlasov-Poisson 1D utilisant les bases d'ondelettes bi-orthogonales splines

Résumé : Dans cet article, nous présentons un algorithme résolvant l'équation de Vlasov-Poisson en utilisant les bases d'ondelettes bi-orthogonales splines à support compact. L'intérêt de ces ondelettes est d'améliorer considérablement la précision des solutions, par rapport aux autres schémas numériques. La méthode utilisée est un splitting de Strang et une méthode semi-lagrangienne pour l'équation de Vlasov, et un solveur de Poisson utilisant uniquement les ondelettes.

Mots-clés : Equation de Vlasov-Poisson, ondelettes bi-orthogonales, methode semi-lagrangienne

1 Introduction

1.1 Physical framework

The study of plasma has been developed since many years in a theoretical point of view (study of the universe, of the Van Allen belts, of nuclear fusion power plants,...) as well as in a practical point of view (plasma display monitors, semi-conductors,...).

Several different methods exist for the description of the evolution of a plasma. A classic one uses the kinetic approach. It introduces a distribution function f, for each kind of particles that is in the plasma (electron and all the kind of ions). This function f represents the density of those particles at time t, at position $\mathbf{x}$ and which have the velocity $\mathbf{v}$. In a classical (non relativistic) plasma submitted to an electro-magnetic field, where all the particles are supposed to not collide, each distribution function f satisfies the Vlasov equation:

$$\partial_t f + \mathbf{v}\nabla_{\mathbf{x}} f + \frac{q}{m}\left(\mathbf{E}(t,\mathbf{x}) + \mathbf{v}\times\mathbf{B}(t,\mathbf{x})\right)\nabla_{\mathbf{v}} f = 0, \tag{1}$$

with $(\mathbf{x},\mathbf{v}) \in \mathbb{R}^d \times \mathbb{R}^d$, where q is the charge of the particle, m its mass, and $\mathbf{E}$ and $\mathbf{B}$ the electric and magnetic fields applied to the plasma. The system is closed by the Maxwell equations.

Several authors have numerically studied equation (1), using several methods:

- Spectral methods: the Vlasov equation is splitted in time, to solve separately the advection with respect to $\mathbf{x}$ and the advection with respect to $\mathbf{v}$. Klimas and Farrel have used Fourier techniques in [12]. Shoucri and Gagné ([14]) or Holloway ([11]) have used Hermite polynomials.

- Methods based on space discretization of the phase space: finite volumes ([8], [9]) or semi-lagrangian methods ([1], [4], [10]).

1.2 Description of the method

Our approach is to use semilagrangian methods to solve the Vlasov-Poisson equations. Several authors have already implemented such numerical schemes (see [4] or [10]) using semilagrangian adaptive methods. This method is based on the fact that the solution of a hyperbolic partial differential equation is constant along its characteristics. If we look for the function at time t_{n+1}, we go backward on the characteristic and find the value at time t_n. This latter is usually found by interpolating the known solution at time t_n. See section 2 for a more detailed description.

The aim of this work is to improve such methods by using wavelets. The interest of wavelets are their great compression capability, and the fact that they have null moments, so that the mass and the quantity of movement are unchanged during compression (which will be a future work). In addition, we will use compact bi-orthogonal spline wavelets, with which we can achieve highly precise methods.

In this paper, we consider a plasma that is submitted to an autoconsistant electro-magnetic field. We set the dimension d to 1. Equation (1) is then coupled with the Poisson equation, and we obtain the so called Vlasov-Poisson equations, which can be written (in a dimensionless form) as:

$$\partial_t f + v\partial_x f - E(t,x)\partial_v f = 0, \tag{2}$$

$$\partial_x E = 1 - \int_{\mathbb{R}} f \mathrm{d}v. \tag{3}$$

For a derivation of the dimensionless form, see Appendix A.

The boundary conditions are the following: f and E are periodic with respect to x, and f tends to 0 when v tends to $\pm\infty$.

1.3 Outline of the paper

This paper is organized as follows:

- In Section 2, we briefly present the Strang splitting and how we use it with the advection equation. It uses the fact that the function is constant along the characteristics of the advection equations.
- As we will choose a wavelet (or more precisely a scale function) for our test function in Section 2, the Section 3 will present the basics of the Multi-Resolution Analysis theory that we need. In particular, we will present the bi-orthogonal wavelets introduced in [6] and a quadrature formula to compute efficiently the inner product of a function with a scale function.
- The Section 4 will describe shortly the implementations of the advection algorithm, and of the Poisson solver. Then we will describe the main algorithm of our scheme.
- In Section 5, we will present the numerical results obtained with our scheme. Two test cases will be performed: the Landau damping one and the two-stream instability one.

2 Method: the Strang splitting

We have used the classical second order in time discretization (the Strang splitting, *cf* [17] and [16]), to separate the computation of each advection. Then the problem is reduced to two one-dimensional advections:

$$\partial_t f + v\partial_x f = 0 \quad (4)$$
$$\partial_t f - E(t,x)\partial_v f = 0 \quad (5)$$

The Strang splitting is the following scheme. If f is known at time $t_n = n\Delta t$ ($f_n = f(t_n)$),

1. We solve (4) on $[t_n, t_{n+\frac{1}{2}}]$ and we get $f_{n+\frac{1}{2}}$ from f_n.
2. We solve (5) on $[t_n, t_{n+1}]$ and we get $\tilde{f}_{n+1}$ from $f_{n+\frac{1}{2}}$.
3. We solve (4) on $[t_{n+\frac{1}{2}}, t_{n+1}]$ and we get f_{n+1} from $\tilde{f}_{n+1}$.

We remark that v is constant on both steps 1) and 3), and x is constant on step 2), so is $E(t_n, x)$. So it is sufficient to solve $\partial_t f + c\partial_z f = 0$ on $[t, t+\Delta t]$, with $c \in \mathbb{R}$ and $f : (t,z) \mapsto f(t,z)$. The velocity c will be respectively v and $-E(t_n,x)$ for the computation of (4) and (5).

If $\omega(t,x)$ is a test function, then $(f,w)(t_{n+1}) = (f,\omega)(t_n)$ with $\partial_t\omega + c\partial_z\omega = 0$ (f is constant along the characteristics of the advection equation) and $\omega(t_{n+1}) = \omega_0$, ω_0 being a function that we will choose later. Then $\omega(t_n, z) = \omega(t_{n+1}, z - c\Delta t) = \omega_0(z + c\Delta t)$. So

$$(f(t_{n+1}), \omega_0) = \int_{\mathbb{R}} f(t_n, z)\omega_0(z + c\Delta t)\mathrm{d}z. \quad (6)$$

Of course, the choice of ω_0 will be a wavelet, or more precisely, a *scale function.*

3 Wavelets basics

During the last two decades, wavelets decomposition has been widely studied, both from theoretical and practical points of view. It gives a sparse expansion of a function of $\mathrm{L}^2(\mathbb{R})$. The multiresolution analysis (MRA), introduced by Y. Meyer and S. Mallat ([13]), is one of the most powerful tools to build wavelets. As well as in Fourier analysis, a fast wavelet transform (FWT), introduced by S. Mallat ([13]) computes the expansion with a complexity of $O(n)$.

In this section, we recall briefly the multiresolution analysis theory (Section 3.1). Then Section 3.2 will present the bi-orthogonal bases of wavelets and its fast wavelet transform. In Section 3.3, we will detail the wavelets used in our scheme: the bi-orthogonal compactly supported spline wavelets. Finally, in Section 3.4, we will present a quadrature formula that computes efficiently inner products of functions of $\mathrm{L}^2(\mathbb{R})$ with scale functions and used for the inner product (6).

3.1 Multiresolution analysis

The aim of the multiresolution analysis is to build an orthonormal basis of $\mathrm{L}^2(\mathbb{R})$, commonly called wavelet basis, composed of translations and dilations of a unique function (the mother wavelet). The expansion of a function of $\mathrm{L}^2(\mathbb{R})$ in this basis has rapidly decreasing components.

Strictly speaking, a MRA of $\mathrm{L}^2(\mathbb{R})$ is a pair $\left((V_j)_{j\in\mathbb{Z}}, \varphi\right)$, where V_j is a closed subspace of $\mathrm{L}^2(\mathbb{R})$ and φ a function of $\mathrm{L}^2(\mathbb{R})$, that satisfies the following properties:

1. $\forall j \in \mathbb{Z}$, $V_j \subset V_{j+1}$ (increasing),
2. $\overline{\cup_{j\in\mathbb{Z}} V_j} = \mathrm{L}^2(\mathbb{R})$ (density),
3. $\cup_{j\in\mathbb{Z}} V_j = \{0\}$ (separability),
4. $\forall j \in \mathbb{Z}$, $f \in V_j \Leftrightarrow f(2\cdot) \in V_{j+1}$ (scaling),
5. $\{\varphi(\cdot - k),\ k \in \mathbb{Z}\}$ is an orthonormal basis of V_0.

Definition 3.1 *The function φ is called the scaling function.*

By noting $\varphi_{j,k} = 2^{\frac{j}{2}}\varphi(2^j \cdot - k)$, we remark easily that $\{\varphi_{j,k},\ k \in \mathbb{Z}\}$ is an orthonormal basis of V_j (from property 4. and 5. above). The set V_j represents the "scale" j of the expansion.

By setting $W_j = V_{j+1} \cap V_j$ (that is, W_j being the orthogonal component of V_j in V_{j+1}, hence $V_{j+1} = V_j \oplus W_j$), we can show (*cf* [3]) that there exists a function ψ in $\mathrm{L}^2(\mathbb{R})$ such that $\{\psi(\cdot - k),\ k \in \mathbb{Z}\}$ is an orthonormal basis of W_0. We may interpret W_j as what lacks to V_j (the information at scale j) to be in V_{j+1} (the information at higher scale $j+1$), that is, a detail.

Definition 3.2 *The function ψ is called the mother wavelet.*

Then, we can easily check that

$$\mathrm{L}^2(\mathbb{R}) = \oplus_{j\in\mathbb{Z}} W_j = V_{J_c} \oplus (\oplus_{j\geqslant J_c} W_j),$$

where J_c is any integer (it represent the coarse scale of the expansion). Hence, any function f of $\mathrm{L}^2(\mathbb{R})$ could be written as

$$\begin{aligned} f &= \sum_{j\in\mathbb{Z}}\sum_{k\in\mathbb{Z}} (f,\psi_{j,k})\,\psi_{j,k}, \\ &= \sum_{k\in\mathbb{Z}} (f,\varphi_{J_c,k})\,\varphi_{J_c,k} + \sum_{j\geqslant J_c}\sum_{k\in\mathbb{Z}} (f,\psi_{j,k})\,\psi_{j,k}. \end{aligned} \tag{7}$$

Remark 3.1 *There is a unique (but simple) way to construct ψ from φ. As $\varphi \in V_0 \subset V_1$, there exists $(h_k)_{k\in\mathbb{Z}}$ such that $\varphi = \sum_{k\in\mathbb{Z}} h_k \varphi_{1,k}$. Then we set $\psi = \sum_{k\in\mathbb{Z}} g_k \varphi_{1,k}$, with $g_k = (-1)^k h_{1-k}$.*

Definition 3.3 *With the notations of the Remark 3.1, we define the filter of a scale function φ as $m_\varphi = \frac{1}{\sqrt{2}} \sum_{k\in\mathbb{Z}} h_k e^{-i\xi k}$. The filter of a mother wavelet is defined in a similar manner, with the coefficients g_k instead of h_k.*

The fast wavelet transform, which links a fine scale to a coarser one, is then easily calculated. If $f \in \mathrm{L}^2(\mathbb{R})$, let note $\alpha_{j,k} = (f,\varphi_{j,k})$ and $\beta_{j,k} = (f,\psi_{j,k})$. Then:

$$\begin{aligned} \alpha_{j,k} &= \sum_{l\in\mathbb{Z}} h_{l-2k}\alpha_{j+1,l}, \\ \beta_{j,k} &= \sum_{l\in\mathbb{Z}} g_{l-2k}\alpha_{j+1,l}. \end{aligned}$$

The inverse wavelet transform does the reverse process:

$$\alpha_{j+1,k} = \sum_{l\in\mathbb{Z}} h_{k-2l}\alpha_{j,l} + \sum_{l\in\mathbb{Z}} g_{k-2l}\beta_{j,l}.$$

If we want to approximate f, we just have to cut the expansion (7) to some fine scale J_f. Then, applying the fast wavelet transform up to the coarse scale J_c leads to:

$$f \simeq \sum_{k\in\mathbb{Z}} \alpha_{J_c,k}\varphi_{J_c,k} + \sum_{j=J_c}^{J_f-1}\sum_{k\in\mathbb{Z}} \beta_{j,k}\psi_{j,k} = \sum_{k\in\mathbb{Z}} \alpha_{J_f,k}\varphi_{J_f,k},$$

which is the approximation of f at the fine scale J_f.

The next section will present a way to uncouple the analysis of the signal (that is, the inner product $\alpha_{j,k}$) and its reconstruction, by introducing the dual functions of φ and ψ, $\tilde{\varphi}$ and $\tilde{\psi}$: the bi-orthogonal wavelets.

3.2 Bi-orthogonal wavelets

Bi-orthogonal wavelets have been introduced in [6] in order to construct wavelets with good properties in image processing (that is, producing less artifacts).

Bi-orthogonal wavelets introduce an additional MRA $\left(\left(\tilde{V}_j\right)_{j\in\mathbb{Z}}, \tilde{\varphi}\right)$ such that $(\varphi, \tilde{\varphi}(\cdot - k)) = \delta_{0,k}$, for all $k \in \mathbb{Z}$. The function $\tilde{\varphi}$ is called the *dual scale function*. We can then construct a mother wavelet ψ and its dual one $\tilde{\psi}$: if $\varphi = \sum_{k\in\mathbb{Z}} h_k \varphi_{1,k}$ and $\tilde{\varphi} = \sum_{k\in\mathbb{Z}} \tilde{h}_k \tilde{\varphi}_{1,k}$, then $\psi = \sum_{k\in\mathbb{Z}} g_k \varphi_{1,k}$ and $\tilde{\psi} = \sum_{k\in\mathbb{Z}} \tilde{g}_k \tilde{\varphi}_{1,k}$, with $g_k = (-1)^k \tilde{h}_{1-k}$ and $\tilde{g}_k = (-1)^k h_{1-k}$.

Then, if $f \in \mathrm{L}^2(\mathbb{R})$, $\tilde{\alpha}_{j,k} = (f, \tilde{\varphi}_{j,k})$ and $\tilde{\beta}_{j,k} = \left(f, \tilde{\psi}_{j,k}\right)$, the fast wavelet transform in that case is:

$$\begin{aligned} \tilde{\alpha}_{j,k} &= \sum_{l\in\mathbb{Z}} \tilde{h}_{l-2k} \tilde{\alpha}_{j+1,l}, \\ \tilde{\beta}_{j,k} &= \sum_{l\in\mathbb{Z}} \tilde{g}_{l-2k} \tilde{\alpha}_{j+1,l}, \end{aligned}$$

and the inverse wavelet transform reads:

$$\tilde{\alpha}_{j+1,k} = \sum_{l\in\mathbb{Z}} h_{k-2l} \tilde{\alpha}_{j,l} + \sum_{l\in\mathbb{Z}} g_{k-2l} \tilde{\beta}_{j,l}.$$

Finally, f can be expanded in the following manner:

$$f \simeq \sum_{k\in\mathbb{Z}} \tilde{\alpha}_{J_c,k} \varphi_{J_c,k} + \sum_{j=J_c}^{J_f-1} \sum_{k\in\mathbb{Z}} \tilde{\beta}_{j,k} \psi_{j,k} = \sum_{k\in\mathbb{Z}} \tilde{\alpha}_{J_f,k} \varphi_{J_f,k}. \tag{8}$$

One of the advantages of such an expansion is that the analysis of f is done by functions that have several null moments. This implies that the coefficients $\tilde{\beta}_{j,k}$ will be small in regions where f is flat.

3.3 Bi-orthogonal compactly supported wavelets

Beyond the fact that they are bi-orthogonal, the interest of such wavelets are multiple. In particular, we can compute the scale functions φ and $\tilde{\varphi}$ explicitly: they are piecewise polynomials, and their coefficients are rational (and whose denominators are a power of 2, which is interesting for numerical computations). They are determined by a pair of integer: the order N and the dual order $\tilde{N}$. The filter m_φ of φ

is given by:

$$m_\varphi(\xi) = \frac{1}{N} \sum_{\lfloor \frac{N}{2} \rfloor}^{N-\lfloor \frac{N}{2} \rfloor} \binom{N}{n + \lfloor \frac{N}{2} \rfloor} e^{-in\xi},$$

where $\lfloor y \rfloor$ is the greatest integer less or equal than y. The filter $m_{\tilde{\varphi}}$ of $\tilde{\varphi}$ is given by:

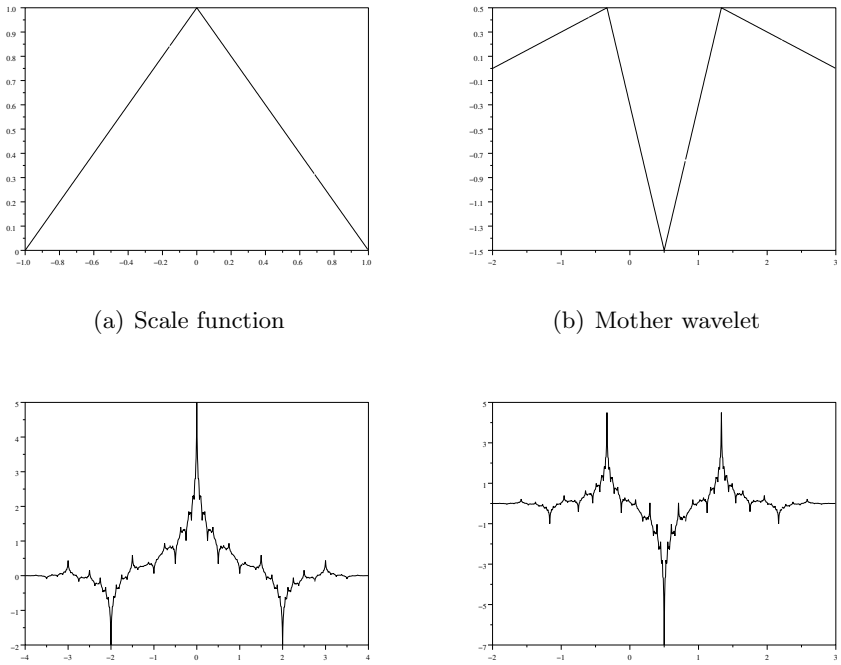
$$m_{\tilde{\varphi}}(\xi) = e^{-\kappa \frac{\xi}{2}} \left(\cos\left(\frac{\xi}{2}\right) \right)^{\tilde{N}} \left(\sum_{n=0}^{k-1} \binom{k-1+n}{n} \left(\sin\left(\frac{\xi}{2}\right) \right)^{2n} \right), \tag{9}$$

where $2k = N + \tilde{N}$ is even and $\kappa = 1$ if N is odd, 0 otherwise. See [6] p. 539 for more details.

We use the notations of section 3.2 for the mother wavelets ψ and $\tilde{\psi}$. It's easy to show that the supports of the filters m_φ, $m_{\tilde{\varphi}}$, m_ψ and $m_{\tilde{\psi}}$ (that is, the support of the sequence of their Fourier coefficients) are compact, hence the supports of φ, $\tilde{\varphi}$, ψ and $\tilde{\psi}$ too. More precisely, we have:

$$\begin{aligned}
\operatorname{supp}(m_\varphi) &= \left\{ -\left\lfloor \frac{N}{2} \right\rfloor, \ldots, N - \left\lfloor \frac{N}{2} \right\rfloor \right\}, \\
\operatorname{supp}(\varphi) &= \left[-\left\lfloor \frac{N}{2} \right\rfloor, N - \left\lfloor \frac{N}{2} \right\rfloor \right], \\
\operatorname{supp}(m_{\tilde{\varphi}}) &= \left\{ -k - \left\lfloor \frac{\tilde{N}}{2} \right\rfloor + 1, \ldots, k + \tilde{N} - \left\lfloor \frac{\tilde{N}}{2} \right\rfloor - 1 \right\}, \\
\operatorname{supp}(\tilde{\varphi}) &= \left[-k - \left\lfloor \frac{\tilde{N}}{2} \right\rfloor + 1, k + \tilde{N} - \left\lfloor \frac{\tilde{N}}{2} \right\rfloor - 1 \right], \\
\operatorname{supp}(m_\psi) &= \left\{ 2 - k - \tilde{N} + \left\lfloor \frac{\tilde{N}}{2} \right\rfloor, \ldots, k + \left\lfloor \frac{\tilde{N}}{2} \right\rfloor \right\}, \\
\operatorname{supp}(\psi) &= \left[\frac{1}{2}\left(2 - N - \tilde{N}\right), \frac{1}{2}\left(N + \tilde{N}\right) \right], \\
\operatorname{supp}\left(m_{\tilde{\psi}}\right) &= \left\{ 1 - N + \left\lfloor \frac{N}{2} \right\rfloor, \ldots, 1 + \left\lfloor \frac{N}{2} \right\rfloor \right\}, \\
\operatorname{supp}\left(\tilde{\psi}\right) &= \left[\frac{1}{2}\left(2 - N - \tilde{N}\right), \frac{1}{2}\left(N + \tilde{N}\right) \right].
\end{aligned}$$

We will choose N and $\tilde{N}$ even, so that φ nd $\tilde{\varphi}$ are even functions. Figure 1 shows the scale ans mother wavelet functions with $N = 2$ and $\tilde{N} = 2$



(a) Scale function

(b) Mother wavelet

(c) Dual scale function

(d) Dual mother wavelet

Figure 1: Scale functions and mother wavelets for order 2 and dual order 2

3.4 Quadrature formula

To compute the coefficients $\alpha_{J_f,k}$ of the expansion of a function f on the fine scale (see Equation (8)), we need to compute the inner product of f and $\tilde{\varphi}$. The problem is that $\tilde{\varphi}$ is not smooth (see Figure 1 c)) and then, standard quadrature formulae do not compute accurately the inner product $(f, \tilde{\varphi})$ unless one takes a large number of points.

In [15], Sweldens and Piessens have introduced several quadrature formulae to compute accurately, and faster that the standard method, the inner product (f, φ) (where φ is any scale function). As we use bi-orthogonal wavelets, the scale function is $\tilde{\varphi}$. The idea is to say that $\tilde{\varphi}$ is located near 0, so that, if f belongs to $\mathrm{L}^2(\mathbb{R})$, $(f, \tilde{\varphi}_{j,k})$ is almost $f\left(\frac{k}{2^j}\right)\tilde{\varphi}(0)$ (that is, we consider that f is constant on the support of $\tilde{\varphi}$). To improve this, we replace f by a Lagrange polynomial interpolation at points $\frac{k+n}{2^j}$, for $n \in \{-\lfloor \frac{d}{2} \rfloor, \ldots, d - \lfloor \frac{d}{2} \rfloor\}$, where $d+1$ is the degree of the polynomial. Then an estimate of $\alpha_{j,k}$ is given by

$$\alpha_{j,k} \simeq \frac{1}{2^{\frac{j}{2}}} \sum_{n=-\lfloor \frac{d}{2} \rfloor}^{d-\lfloor \frac{d}{2} \rfloor} f\left(\frac{k+n}{2^j}\right) \omega_n. \tag{10}$$

The weights ω_n are the inner product of $\tilde{\varphi}$ with the Lagrange polynomial L_n of degree $d+1$ such that $L_n(m) = \delta_{m,n}$, for $n, m \in \left\{-\lfloor \frac{d}{2} \rfloor, \ldots, d - \lfloor \frac{d}{2} \rfloor\right\}$. Hence, to compute the weights ω_n, it is sufficient to compute the moments $M_k = \int_{\mathbb{R}} x^k \tilde{\varphi}(x) \mathrm{d}x$. These moments are easily computed by induction (see [15]):

$$M_k = \frac{1}{2(2^k - 1)} \sum_{l=0}^{k-1} \left(\sum_{n \in \mathbb{Z}} h_n \binom{k}{l} n^{k-l} \right) M_l.$$

Another interesting point is that we only need the values of f at points $\frac{k}{2^j}$ and the coefficients of the filter of $\tilde{\varphi}$, that are explicit rational numbers (see (9)). Then, the weights ω_n are rational numbers and can be explicitly computed. For more details, see [5].

The precision of these quadrature methods depends on several parameters: the discretization, the order and dual order of the dual scale function, and the degree of the polynomial. The table 1 resumes the relative L^2 error between $f(x) = e^{-80(x-0.5)^2}$, $x \in [0,1]$, and the reconstructed function $\tilde{f}$. That is, we compute the inner products $\alpha_{j,k}$ with $j = 7$ thanks to the quadrature formula (10) and we compute $\tilde{f}$ from these inner products thanks to equation (8). This last computation is

	Order-Dual order				
d	(2-2)	(2-4)	(4-2)	(4-4)	(6-2)
3	7.032E-4	7.032E-4	6.736E-6	6.736E-6	1.311E-5
5	7.029E-4	7.055E-4	2.864E-7	2.864E-7	1.334E-7
7	7.029E-4	7.055E-4	3.164E-7	3.422E-7	1.558E-9
9	7.029E-4	7.055E-4	3.165E-7	3.425E-7	2.56E-10

Table 1: Relative error between f and $\tilde{f}$ in L^2 norm for the scale 7

	Order-Dual order				
d	(2-2)	(2-4)	(4-2)	(4-4)	(6-2)
3	1.42s	1.44s	1.46s	1.43s	1.44s
5	1.80s	1.82s	1.82s	1.82s	1.82s
7	2.18s	2.21s	2.20s	2.21s	2.25s
9	2.50s	2.59s	2.57s	2.58s	2.58s

Table 2: Time (in seconds) to compute 1E4 times $\tilde{f}$ from f

exact as $\varphi_{j,k}$ is a piecewise polynomial (see Section 3.3). So the measurement of this error is the error of the quadrature formula.

From Table 1, we see that choosing an order of 6 gives the best results. Increasing the dual order does not seem to improve the precision of the quadrature formula. Using a higher degree for the interpolation improves the precision a lot if the order is 4 or 6. Using a higher order does not increase much the precision (if order is equal to 8 and if d is equal to 9, the error is also of order 1E-10). Table 2 shows the time to compute 10000 times $\tilde{f}$ from f. Of course, increasing the order and the degree decreases the speed, but the computations are fast. These computations have be performed on a Pentium 3 1Ghz. The program is written in Fortran 90.

4 Implementation

In this section, we shortly describe the different parts of the implementation of the algorithm: the advection algorithm and the Poisson solver. We also present briefly the main algorithm, which is quite straightforward.

4.1 Advection on finest scale

In this section, we describe briefly the implementation for the advections with respect to x and v.

We suppose that we know $\alpha_{j,k}(t) = (f(t), \tilde{\varphi}_{j,k})$, where j is the scale, k a integer, f the function and t the time. f is supposed to be compactly supported. The time step is denoted by Δt. We want to compute $\alpha_{j,k}(t+\Delta t)$. We use (6) with $\omega_0 = \tilde{\varphi}_{j,k}$. So

$$\alpha_{j,k}(t + \Delta t) = \int_{\mathbb{R}} f(t,x)\tilde{\varphi}_{j,k}(x + c\Delta t)\mathrm{d}x.$$

As, $f(t,\cdot) = \sum_{l\in\mathbb{Z}} \alpha_{j,l}(t)\varphi_{j,l}$,

$$\alpha_{j,k}(t + \Delta t) = \sum_{l\in\mathbb{Z}} \alpha_{j,l}(t) \int_{\mathbb{R}} \varphi_{j,l}(x)\tilde{\varphi}_{j,k}(x + c\Delta t)\mathrm{d}x.$$

The sum over l and the integral may commute as all the functions are compactly supported and the sum over l is finite.

Then we integrate by parts so that

$$\alpha_{j,k}(t + \Delta t) = \sum_{l\in\mathbb{Z}} \alpha_{j,l}(t) \int_{\mathbb{R}} \varphi_{j,l}(x - c\Delta t)\tilde{\varphi}_{j,k}(x)\mathrm{d}x.$$

The reasons are that $\tilde{\varphi}_{j,k}$ is not smooth, while $\varphi_{j,l}$ is, and that we know $\varphi_{j,l}$ exactly, as it is a piecewise polynomial. Hence we can use the quadrature method (10) to compute this integral. Finally;

$$\begin{aligned} \alpha_{j,k}(t + \Delta t) &= \sum_{l\in\mathbb{Z}} \alpha_{j,l}(t) \sum_{n=-\lfloor \frac{d}{2} \rfloor}^{d-\lfloor \frac{d}{2} \rfloor} \varphi(k + n - l - 2^j c\Delta t)\omega_n, \\ &= \sum_{l\in\mathbb{Z}} \alpha_{j,l}(t) c_{j,k,l}(t). \end{aligned} \tag{11}$$

- In the case of the advection with respect to x, the velocity c is equal to v (see 2). As the phase space is uniformly discretized, one can compute one and for all the coefficients $c_{j,k,l}(t)$ at the beginning of the program. Thus, (11) is simply a multiplication matrix-vector.
- In the case of the advection with respect to v, the velocity c is equal to the electric field $E(t,\cdot)$ (see 2). So we can not use the same trick as for the

	Order-Dual order	
Fine scale	(4-2)	(6-2)
6	8.18E-4	1.95E-5
7	3.16E-7	3.25E-8

Table 3: Advection with respect to x: relative error

	Order-Dual order	
Fine scale	(4-2)	(6-2)
6	0.03s	0.04s
7	0.18s	0.18s

Table 4: Advection with respect to x: computation time

advection with respect to x. But we compute only the coefficients that are non zero (that is, we choose k and l such that the interior of $\mathrm{supp}(\varphi_{j,l}) \cap \mathrm{supp}(\tilde{\varphi}_{j,l})$ is empty. Nevertheless, this computation is not as fast as the computation of the advection with respect to x.

As an example, we solve

$$\partial_t f + \frac{1}{2}\partial_x f = 0$$

on $[0,T] \times [0,1]$, with initial condition $f_0 = e^{-80(x-\frac{1}{2})^2}$, with $\Delta t = 0.01$, with 200 iterations (that is, we iterate over one period), we have the following results:

Table 3 shows that the precision is again the best with the order and dual order equal to 6 and 2, with respect to 4 and 2, which is not surprising. Also, the precision with a fine scale equal to 6 is far below the fine scale 7. Of course, the computation time is lower with the fine scale equal to 6 than with 7 (*cf* Table 4).

4.2 Poisson solver

In this section, we present a simple way to solve accurately the Poisson equation in one dimension, using the wavelets. As we work only with the scale function, this method only use the identity

$$\begin{aligned} f(x,v) &= \sum_{k\in\mathbb{Z}} \left(\sum_{l\in\mathbb{Z}} \left(f, \tilde{\varphi}_{j,k}\right), \tilde{\varphi}_{j,l} \right) \varphi_{j,k}(x)\varphi_{j,l}(v), \\ &= \sum_{k\in\mathbb{Z}} \sum_{l\in\mathbb{Z}} c_{j,k,l}\varphi_{j,k}(x)\varphi_{j,l}(v), \end{aligned} \tag{12}$$

where j is the used scale. More precisely, we want to solve

$$\partial_x E(t,\cdot) = 1 - \int_{\mathbb{R}} f(t,\cdot,v)\mathrm{d}v. \tag{13}$$

We plug (12) into (13) and we use the fact that $\int_{\mathbb{R}} \varphi_{j,l}(v)\mathrm{d}v = 2^{-\frac{j}{2}}$. We get

$$\begin{aligned} E(x) &= E(0) + \frac{1}{2^{\frac{j}{2}}} \sum_{k\in\mathbb{Z}} \sum_{l\in\mathbb{Z}} c_{j,k,l} \int_0^x \varphi_{j,k}(s)\mathrm{d}s, \\ &= E(0) + \frac{1}{2^j} \sum_{k\in\mathbb{Z}} \sum_{l\in\mathbb{Z}} c_{j,k,l} \int_{-k}^{2^j x - k} \varphi(s)\mathrm{d}s. \end{aligned}$$

The value of $\int_{-k}^{2^j x-k} \varphi(s)\mathrm{d}s$ is computed explicitly, as φ is a piecewise polynomial.

If we apply our solver to the function $f = e^{-80v^2}\sin(2\pi x)$, with $J_f = 7$, $(N, \tilde{N}) = (6, 2)$, then, the relative error is about 1E-14, that is, the reconstruction is perfect. Also, this solver is only a product matrix-vector, and then is fast : it takes 2s to resolve 1E3 times the Poisson equation.

4.3 Description of the algorithm

The algorithm is quite straightforward:

1. We choose J_f, Δt, a class of wavelets and an initial condition $f \in \mathrm{L}^2(\mathbb{R})$.
2. We compute the coefficients $\alpha_{J_j,k} = (f, \tilde{\varphi}_{J_f,k})$.
3. We solve the Vlasov equation thanks to the Strang splitting:
   - $\partial_t f + v\partial_x f = 0$ over a time interval of $\frac{\Delta t}{2}$.
   - $\partial_t f + E\partial_v f = 0$ over a time interval of Δt.
   - $\partial_t f + v\partial_x f = 0$ over a time interval of $\frac{\Delta t}{2}$.
4. We solve the Poisson equation.
5. We loop to 3.
6. We compute f with the scale function and the coefficients $\alpha_{J_j,k}$.

5 Numerical results

In this section, we present the numerical results obtained with our algorithm to solve the Vlasov-Poisson equations (2)-(3). We have chosen classical test cases, that is, the Landau damping and two-stream instability test cases. The numerical parameters that we use are the following:

- The domain on which we solve the Vlasov-Poisson equation (in its dimensionless form) is $[0, L] \times [-V, V]$, where $L = 4\pi$ and $V = 10$.
- We have performed the computations with time steps equal to 0.1 and 0.01. The time interval is 40, that is, 40 times the plasma pulsation.
- The fine scales J_f used for the wavelet decomposition are 6 and 7. That is, the domain $[0, L] \times [-V, V]$ is discretized with 2^{J_f} with respect to x and with 2^{J_f+1} with respect to v. The reason why the number of points with respect to v is twice as high as in the x direction is that it gives more precise computations.
- Finally, the tests done in section 3.4 and in the test cases lead to the same conclusion: the order and dual order equal to respectively 6 and 2, with a degree of 9 for the computation of the quadrature formula give the best results and are sufficiently fast. These values are chosen for both test cases.

5.1 Landau damping

The Landau damping phenomenon is an energy exchange between the particles that have a velocity close to the phase velocity, and the electric field. The latter gives a part of its energy to the particles and its energy decays exponentially. In a logarithmic scale, it decays linearly.

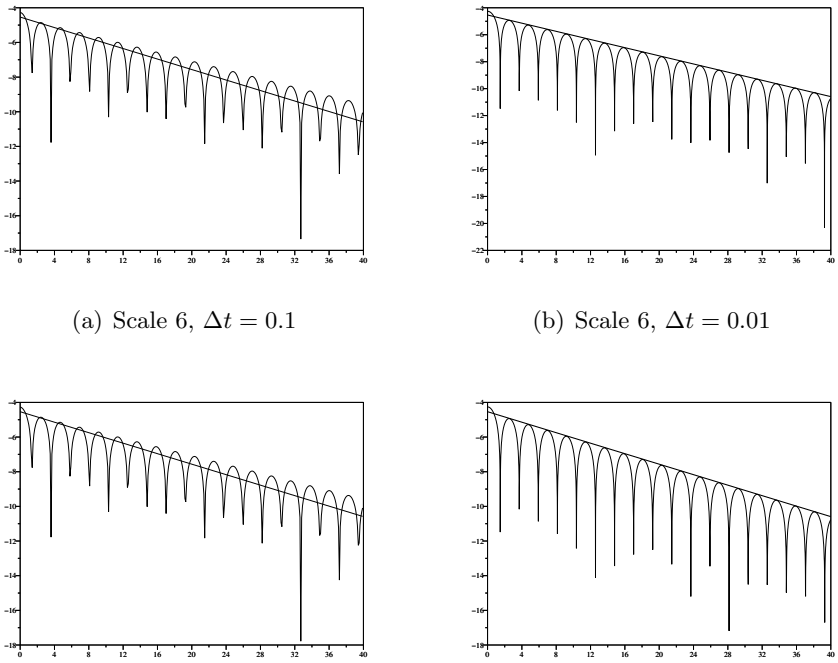
To obtain this behavior, we choose a small perturbation of a maxwellian (which is an obvious steady solution of the Vlasov-Poisson equations) as the initial condition:

$$f_0(x, v) = \frac{1}{\sqrt{2\pi}} e^{-\frac{v^2}{2}} \left(1 + \epsilon \cos(kx)\right),$$

with $k = 0.5$ and $\epsilon = 0.01$.

The theoretical Landau damping is given by $\gamma(k) = \sqrt{\frac{\pi}{8}} k^{-3} e^{-\frac{k^{-2}+3}{2}}$, that is, $\gamma(0.5) \simeq 0.15138$.

Figures 2(a), (b), (c) and (d) represent the evolution of the electric energy (the square of its L^2-norm) in a logarithm scale, obtained with a time step equal to 0.1



(a) Scale 6, $\Delta t = 0.1$

(b) Scale 6, $\Delta t = 0.01$

(c) Scale 7, $\Delta t = 0.1$

(d) Scale 7, $\Delta t = 0.01$

Figure 2: Landau damping

	Scale	
Time step	6	7
0.1	0.05	0.05
0.01	0.05	0.05

Table 5: Mass of the distribution function f

	Scale	
Time step	6	7
0.1	1E-6 / 6E-6	1E-6 / 6E-6
0.01	1E-6 / 6E-6	1E-6 / 6E-6

Table 6: Variations of the L^2 norm of the distribution function f. Value: 0.11876

	Scale	
Time step	6	7
0.1	0 / 2E-4	0 / 2E-4
0.01	0 / 2E-4	0 / 2E-4

Table 7: Variations of the energy of the distribution function f. Value: 5E-4

and 0.01, and a fine scale equal to 6 and 7 (Figures 2 (b) and (d)). The line displayed on the figures has a decay rate equal to 0.15138.

When the time step is equal to 0.01, we see that the damping rate of the energy of the electric field is very close to the theoretical one. In addition, the frequency of the oscillations is about 1.4093, while the theoretical frequency is 1.415.

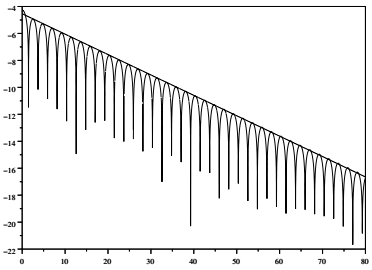
But when the time step is equal to 0.1 (Figures 2 (a) and (c)), the decay rate is not good anymore. It seems that this method does not work on large time steps, although we don't know why we have this behavior. It does not seem to be a precision problem, as it remains when we choose a scale of 8.

In this test case, the mass, the L^2-norm, the total energy and the entropy should be conserved. The mass is really conserved and is constant and remains equal to 0.05 during the computation (Table 5). The Table 6, which is about the L^2-norm of the distribution function, shows that the relative error is about 1E-5. In Table 7, the relative error is about 1E-2, but the total energy oscillates and tends quickly to the value 0.05. Finally, the kinetic entropy is around -0.07 and the relative error is about 1E-3 (see Table 8)

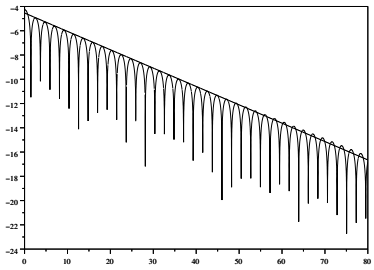
	Scale	
Time step	6	7
0.1	-5.3E-5 / -4.6E-5	-5.2E-5 / -4.6E-5
0.01	-5.2E-5 / -4.6E-5	-5.2E-5 / -4.6E-5

Table 8: Variations of the entropy of the distribution function f. Value: -0.0709

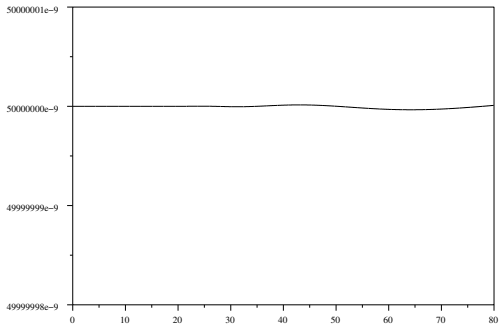
The algorithm is quite stable in that test case. If we do the computations up to a time of 80, the decay rate remains very good (Figure 5.1 (a) and (b)). In addition, in the worst case (scale 6), the mass is quite well conserved, the relative error being less than 1E-8 (Figure 5.1 (c)).



(a) Scale 6, $\Delta t = 0.01$



(b) Scale 7, $\Delta t = 0.01$



(c) Conservation of the mass, scale 6

5.2 Two-stream instability

The two-stream instability is a particles trap phenomenon. It occurs when one perturbs a two-stream steady solution of the Vlasov-Poisson equation:

$$f_0(x,v) = \frac{1}{\sqrt{2\pi}} v^2 e^{-\frac{v^2}{2}} \left(1 + \epsilon \cos\left(Lkx\right)\right).$$

The coefficients k and ϵ are equal to 0.5 and 0.01 respectively. Figure 3 shows the distribution function at time 0. Numerically, we should observe the creation of a

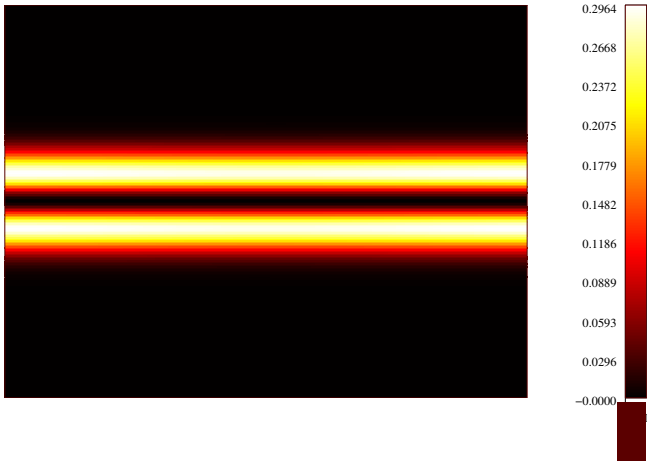
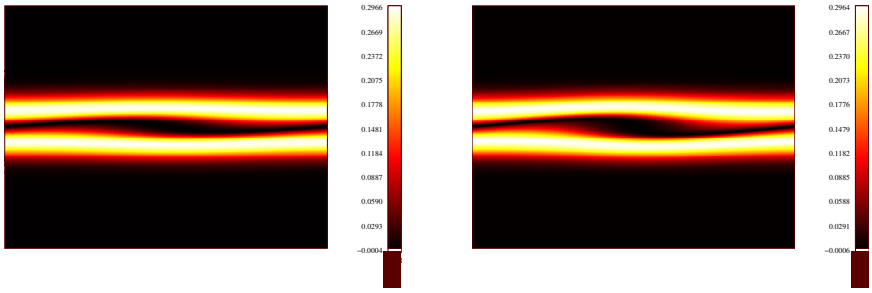


Figure 3: Distribution function at time 0, scale 7

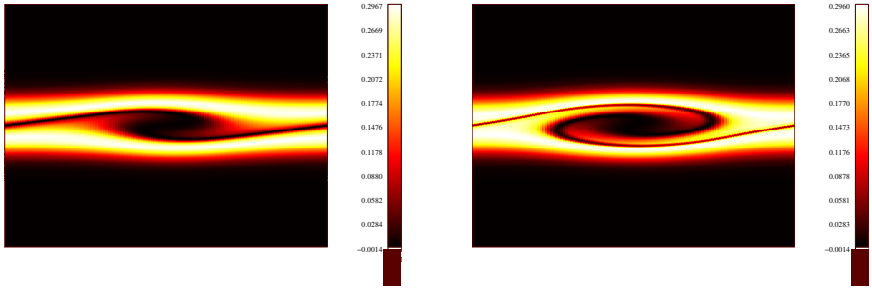
vortex between the two streams. Figures 4(a), (b), (c) and (d) show the creation of the vortex at times 15, 17, 20 and 25, with a fine scale equal to 7 and a time step equal to 0.1. The vortex is formed and this behavior is similar to the result of other numerical scheme ([2]). When the fine scale is equal to 6, the results are similar.

A more interesting case is shown in Figure 5, when the fine scale is equal to 7 and the time step to 0.01. Then, at time 25, we can see micro-structures that appear inside the vortex. This is the filamentation phenomenon. Hence, our scheme can track these micro-structures.

We can also remark that the distribution function is positive, except in some small regions inside the vortex. This is due to the computations involved in the



(a) $t = 15$ (b) $t = 17$



(c) $t = 20$ (d) $t = 25$

Figure 4: Distribution function, scale 7, $\Delta t = 0.1$

quadrature formula computations: the lagrange polynomials oscillate around the edges of the integration domain.

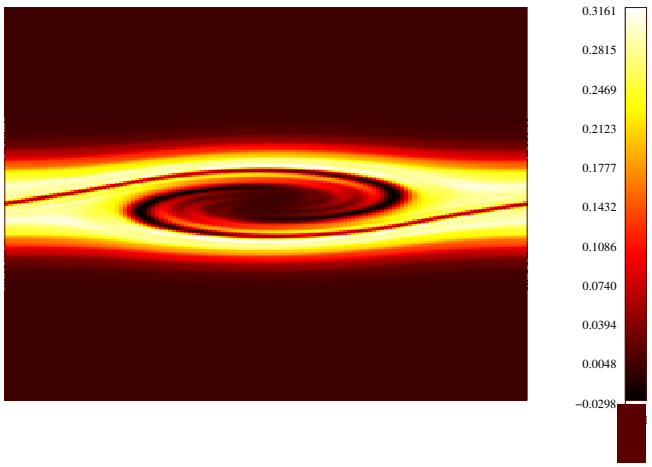


Figure 5: Distribution function, scale 7, $\Delta t = 0.01$, at time 25

6 Conclusion

In this paper, we have presented a numerical scheme based on bi-orthogonal compactly supported spline wavelets. One of their main interest is that they allow precise computations, when their order and dual order are correctly chosen. Nevertheless, this scheme seems to give bad results for the Landau damping when the time step is too high.

The counterpart of its precision is that it is slower than some other existing numerical scheme. One way to improve the speed is to use the 2D ELLAM scheme for the Vlasov equation.

The other main interest of the wavelets is that they compress data quite efficiently. As our main goal is to simulate the Vlasov-Poisson in higher dimensions, the memory used to store the data becomes huge (up to 10 Gigabytes in the three-dimensional case). Hence an adaptive numerical scheme, based on our numerical

scheme has been begun. This can lead to a compression of the data to at most 90 percent.

Acknowledgment I would like to thank Professor Eric Sonnendrücker for his advices and for his interest in my work, and Professor Pierre Bertrand for his wonderful discussions about physics modelization and plasma physics.

A Physical modelization

Let consider a plasma with electrons (with charge $-e$) and ions (with charge e). The ions are supposed to be motionless. Let m be the mass of one electron. Then, the distribution function of the electrons is given by the Vlasov equation

$$\partial_t f(t,\mathbf{x},\mathbf{v}) + \mathbf{v}\nabla_{\mathbf{x}} f(t,\mathbf{x},\mathbf{v}) - \frac{e}{m}\mathbf{E}(t,\mathbf{x})\nabla_{\mathbf{v}} f(t,\mathbf{x},\mathbf{v}) = 0,$$

for $(t,\mathbf{x},\mathbf{v}) \in \mathbb{R}\times[0,L\lambda_D]^d\times\mathbb{R}^d$, where $d\in\mathbb{N}^*$, λ_D is the Debye length, L a dimensionless parameter and $\mathbf{E}$ is the electric field. The distribution function f and the electric field $\mathbf{E}$ are periodic with respect to $\mathbf{x}$, with period $L\lambda_D$. The Poisson equation reads

$$\operatorname{div}(\mathbf{E}) = \frac{1}{\epsilon_0}\rho,$$

where ϵ_0 is the permittivity of the vacuum and ρ is the charge density. By definition,

$$\rho(t,\mathbf{x}) = -en_e(t,\mathbf{x}) + en_i,$$

where n_i is the ion density and n_e the electron density. The electron density is given by $n_e(t,\mathbf{x}) = \int_{\mathbb{R}^d} f(t,\mathbf{x},\mathbf{v})\,\mathrm{d}\mathbf{v}$, by definition of f. The neutrality of the plasma implies that $\int_{[0,L\lambda_D]^d}\rho(t,\mathbf{x})\mathrm{d}\mathbf{x} = 0$.

Let us restrict now to the one dimensional case (d=1). The neutrality of the plasma implies that E is also periodic with respect to x. Hence, the Vlasov-Poisson in 1D case reads as follows:

$$\left\{\begin{array}{rcl} \partial_t f + v\partial_x f + \frac{e}{m}E(t,x)\partial_v f &=& 0,\\ \partial_x E &=& \frac{e}{\epsilon_0}\int_{\mathbb{R}} f\mathrm{d}v - \frac{en_0}{\epsilon_0}. \end{array}\right. \tag{14}$$

The distribution function f is defined on $\Omega = [0,L\lambda_D]\times\mathbb{R}$. For numerical reasons, Ω is constrained to $[0,L\lambda_D]\times[-Vv_{\mathrm{th}},Vv_{\mathrm{th}}]$, where v_{th} is the thermal velocity and V a dimensionless parameter. As we have said, f and E are $L\lambda_D$-periodic with respect to x, and f vanishes at $v=-Vv_{\mathrm{th}}$ and Vv_{th}.

For the Landau damping test, we use the following initial condition:

$$f_0(x,v) = \frac{1}{\sqrt{2\pi}v_{\text{th}}}\left(1+\epsilon\cos\left(kx\right)\right)e^{-\frac{1}{2}\frac{v^2}{{v_{\text{th}}}^2}},$$

where ϵ is a small parameter and k is chosen so that the damping occurs, that is, $k\lambda_D > 0.2$ and $kL\lambda_D = 2\pi$.

We now put (14) in a dimensionless form so that the definition domain with respect to x is $[0,1]$ and the one with respect to v is $[-1,1]$. We note

$$\begin{aligned} t &= \frac{1}{\omega_p}\tilde{t}, \\ x &= L\lambda_D\tilde{x}, \\ v &= Vv_{\text{th}}\tilde{v}, \\ E(t,x) &= \overline{E}\tilde{E}(\tilde{t},\tilde{x}), \\ f(t,x,v) &= \overline{f}\tilde{f}(\tilde{t},\tilde{x},\tilde{v}), \end{aligned}$$

with ω_p being the pulsation of the plasma. Then, the Vlasov-Poisson equation reads

$$\left\{\begin{array}{rcl} \omega_p\partial_t\tilde{f} + \frac{Vv_{\text{th}}}{L\lambda_D}\tilde{v}\partial_{\tilde{x}}\tilde{f} - \frac{\overline{E}e}{mVv_{\text{th}}}E\partial_{\tilde{v}}\tilde{f} &=& 0, \\ \frac{\overline{E}}{L\lambda_D}\partial_{\tilde{x}}\tilde{E} &=& \frac{e}{\epsilon_0}\left(n_0 - \overline{f}Vv_{\text{th}}\int_{-1}^{1}\tilde{f}\mathrm{d}\tilde{v}\right). \end{array}\right. \tag{15}$$

The initial condition also reads

$$f_0(x,v) = \overline{f}_0\tilde{f}_0 = \frac{1}{\sqrt{2\pi}v_{\text{th}}}\left(1+\epsilon\cos\left(Lk\lambda_D\tilde{x}\right)\right)e^{-\frac{1}{2}V^2\tilde{v}^2}.$$

By definition, the plasma pulsation satisfied $\omega_p = \frac{v_{\text{th}}}{\lambda_D}$, and we choose $\overline{E}$, $\overline{f}_0$ and $\overline{f}$ such that

$$\overline{E} = \frac{mv_{\text{th}}\omega_p}{e}, \tag{16}$$

$$\begin{aligned} \overline{f}v_{\text{th}} &= n_0, \\ \overline{f}_0 v_{\text{th}} &= 1, \end{aligned}$$

$$\frac{en_0\lambda_D}{\epsilon_0\overline{E}} = 1. \tag{17}$$

Remark that if we combine equations (16) and (17), we obtain $\omega_p = \frac{n_0e^2}{m\epsilon_0}$, which the definition of ω_p. Hence, this set of equation is consistent.

We note $\tilde{f}(\tilde{t},\tilde{x},\tilde{v}) = f(t,x,v)$ and $\tilde{E}(\tilde{t},\tilde{x}) = E(t,x)$ (to get rid of the tilde notation). Then, the Vlasov-Poisson in the one dimensional case reads:

$$\left\{ \begin{array}{rcl} \partial_t f + \frac{V}{L} v \partial_x f - \frac{1}{V} E(t,x) \partial_v f & = & 0, \\ \partial_x E & = & L\left(1 - V \int_{\mathbb{R}} f \mathrm{d}v\right), \end{array} \right. \tag{18}$$

the initial condition being

$$f_0(x,v) = \frac{1}{\sqrt{2\pi}} \left(1 + \epsilon \cos\left(Lk\lambda_D x\right)\right) e^{-\frac{1}{2}V^2 v^2},$$

with $(t,x,v) \in \mathbb{R} \times [0,1] \times [-1,1]$ and f and E 1-periodic with respect to x.

References

[1] Nicolas Besse, *Convergence of a semi-Lagrangian scheme for the one-dimensional Vlasov-Poisson system*, SIAM J. Numer. Anal., **42**, no.1, p. 350-382, 2004.

[2] Nicolas Besse, *Etude mathématique et numérique de l'équation de Vlasov non linéaire sur des maillages non structurés de l'espace des phases*, Phd Thesis, University of Strasbourg (France), 2003.

[3] George Bachman, Lawrence Narici and Edward Beckenstein, *Fourier and Wavelet analysis*, Universitext, Springer-Verlag, New-York, 2000.

[4] Nicolas Besse and Eric Sonnendrücker, *Semi-Lagrangian schemes for the Vlasov equation on an unstructured mesh of phase space*, J. Comput. Phys., **191**, no.2, p. 341-376, 2003

[5] Albert Cohen, *Numerical analysis of wavelet methods*, Stidies in Mathematics and its applications, North-Holland Publishing Co. Amsterdam, 2003.

[6] Albert Cohen, Ingrid Daubechies and J.-C. Feauveau, *Bi-orthogonal bases of compactly supported wavelets*, Comm. Pure Appl. Math., **45**, no.5, p. 485-560, 1992.

[7] Richard E. Ewing, Jiangguo Liu and Hong Wang, *Adaptive bi-orthogonal spline schemes for advection-reaction equations*, J. Comput. Phys., **193**, no.1, p. 21-39, 2004.

[8] Francis Filbet, *Convergence of a finite volume scheme for the Vlasov-Poisson system*, SIAM J. Numer. Anal., **39**, no.4, p. 1146-1169, 2001.

[9] Francis Filbet and Eric Sonnendrücker, *Comparison of Eulerian Vlasov solvers*, Comput. Phys. Comm., **150**, no.3, p. 247-266, 2003.

[10] M. Gutnic, M. Haefele, I. Paun and E. Sonnendrücker, *Vlasov simulations on an adaptive phase-space grid*, Comput. Phys. Comm., **164**, no.1-3, p. 214-219, 2004.

[11] James Holloway, *Spectral velocity discretizations for the Vlasov-Maxwell equations*, Transport Theory Statist. Phys., **25**, no.1, p. 1-32, 1996.

[12] A. J. Klimas and W. M. Farrell, *A splitting algorithm for Vlasov simulation with filamentation filtration*, J. Comput. Phys., **110**, no.1, p. 150-163, 1994.

[13] Stéphane Mallat, *Multiresolution approximations and wavelet orthonormal bases of $L^2(R)$*, Trans. Amer. Math. Soc., **315**, no.1, p. 69-87, 1989.

[14] Magdi Shoucri and Réal Gagné, *Numerical solution of a two-dimensional Vlasov equation*, J. Comput. Phys., **25**, no.2, p. 94-103, 1977.

[15] Wim Sweldens and Robert Piessens, *Quadrature formulae and asymptotic error expansions for wavelet approximations of smooth functions*, SIAM J. Numer. Anal., **31**, no.4, p. 1240-1264, 1994.

[16] Bruno Sportisse. *An analysis of operator splitting techniques in the stiff case.* J. Comput. Phys., 161(1):140-168, 2000.

[17] Gilbert Strang. *On the construction and comparison of difference schemes.* SIAM J. Numer. Anal., 5:506-517, 1968.

Unité de recherche INRIA Lorraine
LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399